Bee 2.0 and Wire Cell Toolkit Future Plans

Brett Viren

Physics Department

BROOKHAVEN NATIONAL LABORATORY

Wire Cell Summit 7-9 Dec 2015

Outline

Bee 2.0
- Concepts
- Design

Toolkit
- Roadmap
- Tasks

Summary

Assumptions

Want a tool to inject ad-hoc, human invented heuristics into the reconstruction.

Assumptions:

- → current automated algorithms are not fully adequate.
- → improving them by writing code is too slow.
- → want some full reconstruction results quickly.
- → can "port" human heuristics to new automated code.

Analogy

- Doctors get images from X-ray, CT scan, MRI scans, etc.
- Researchers build an automated computer program to tell what diseases the patient has, based on the features seen in the images.
  - False positives might be high
  - False negatives might be high
  - Hospitals can not afford to have high probability of mistakes
- In the end still need expert doctors consultations to make final decisions

What are the goals?

What we have now:

- Very robust 3-D images.
- View images by human with Bee 1.0.

The current steps:

- Automated pattern recognition (of some efficacy).
- Views of the resulting objects (eg clusters)

Questions:

→ What are the metrics by which to judge the performance?

→ What is the goal to reach

→ How can we improve the performance to reach the goal?

Human vs. Machine

Humans are very good at pattern recognition

- → By eye, can easily judge if and where the reconstruction went wrong.
- → Can we inject such kind of knowledge into the reconstruction?

Machines are very good at processing a lot of data and following our instructions.

- → Can we help the machine to learn?

Human-directed automatic reconstruction

1. Human operates on **selectable graphical elements**, eg:
   - Add or remove cells from a blob or cluster.
   - Join or break clusters or tracks.
   - Change pattern recognition category.
   - Associate clusters to particles (eg, two showers to a π^0)
2. Bee sends *command object* to backend.
3. Database records command and associated initial state.
4. Backend applies command via Wire Cell and reruns automated reconstruction as needed.
5. Database records new state, associated with command.
6. Notify user when done:
   - Bee refreshes event if results are fast, or send email/SMS with new link.

"Physics PhotoShop"

Mining Heuristics

Command database has secondary use:

- Allows for full reproduction/replay of operations later.
- Further, can we mine database for things to feed back to automated code?

→ Frequent human operations may indicate a heuristic which is ripe for invention.

- Maybe this metadata is useful input into some machine learning (suggested to us)

"Google Analytics"

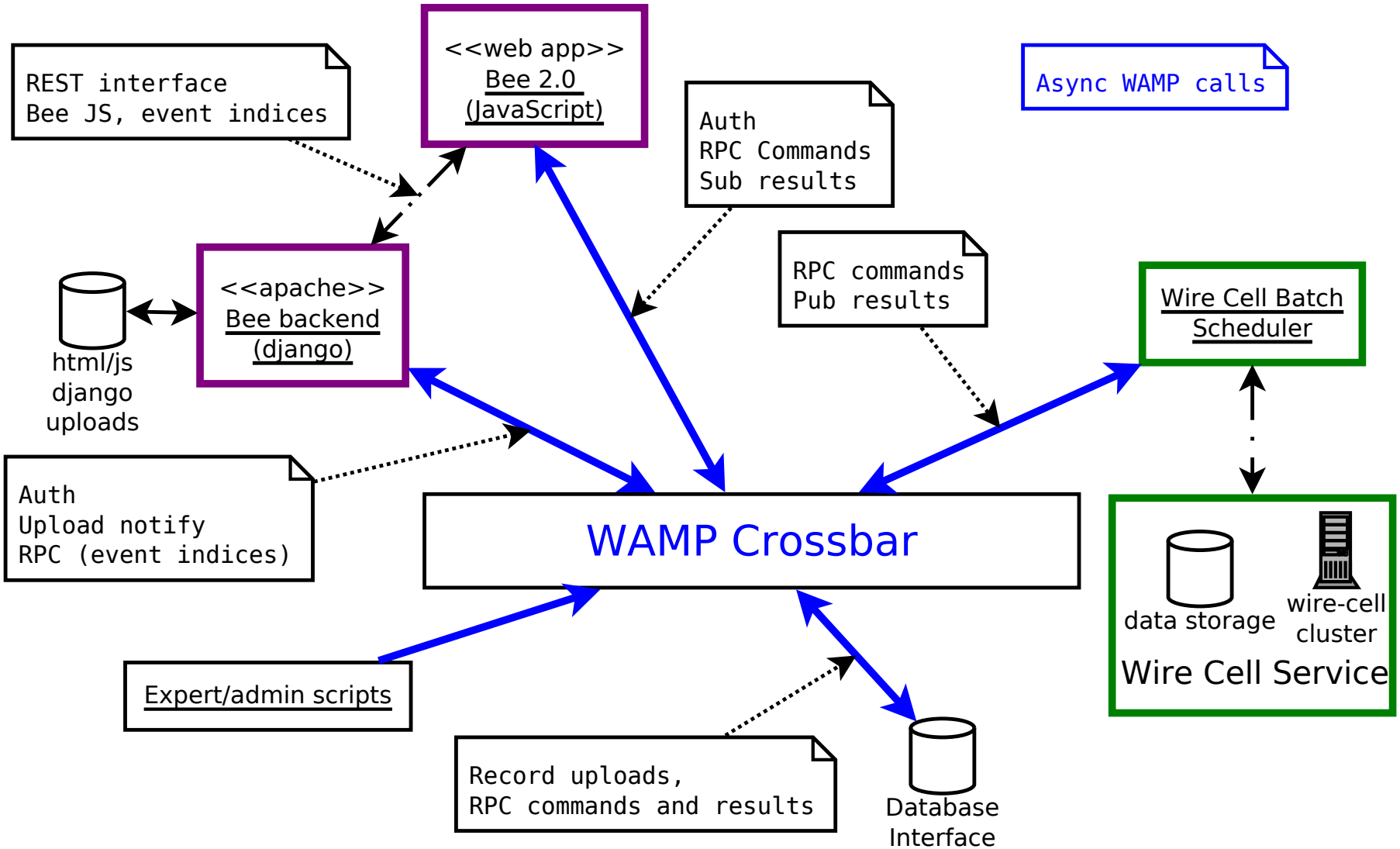
Bee 2.0
Concepts
Design

Toolkit

Summary

Front-end requirements

- Show necessary diagnostic plots/summaries to help the human make decisions.
- Pick/select possible objects
  - Hits, clusters, tracks, vertices, etc.
- Form possible operational command objects
  - Add, delete, split, merge, etc.
- Send operation states to a service.
- Get new results back from the service.
- Live-update the new results in the web browser.
- Support Undo / Revert.


REST interface
Bee JS, event indices
<<web app>>
Bee 2.0
(JavaScript)
Auth
RPC Commands
Sub results
Async WAMP calls
<<apache>>
Bee backend
(django)
html/js
django
uploads
RPC commands
Pub results
Wire Cell Batch
Scheduler
Auth
Upload notify
RPC (event indices)
WAMP Crossbar
data storage
wire-cell
cluster
Wire Cell Service
Expert/admin scripts
Record uploads,
RPC commands and results
Database
Interface

Bee 2.0 Design Highlights

Structure overall system async Pub/Sub & RPC (eg, WAMP).

- A networked crossbar with support for JS, Python, etc.
- Provides flexible ways to return possibly high latency results.
- Assures capture of all input and results.
- Focus on overall system logic not gory details of protocols.

Bee frontend:

- Support more reconstructed object types (clusters, tracks, showers)
- Add selection of entities, forming of command objects.
- User registration UI.
- WAMP hooks (command object/results/auth).

Bee backend

- May still keep django for serving Bee JS, event indices, handling user uploads.

Wire Cell batch service

- Respond to RPC for applying commands.
- Manage job queue running on dedicated cluster.
- Publish results as they come in.

Bee 2.0

Toolkit
- Roadmap
- Tasks

Summary

Wire Cell Toolkit Roadmap

1. Complete design and initial implementation for Data Flow Programming (DFP) execution model based on Intel TBB.
2. Port prototype algorithms into toolkit.
3. Evaluate performance on single-machine, multi-thread.
4. Investigate GPU-accelerated DFP nodes for bottlenecks.
5. Develop a Wire Cell service application.
6. Implement distributed DFP (ie MPI/ZMQ-edges).
7. Port toolkit to run on HPC (target ANL's Mira)

Once #1 is done the rest can proceed, more or less, in parallel.

Volunteers?

I'll go through what each task needs $\longrightarrow$

Finish DFP

- There is a clash between OO and GP to contend with.
- Dynamic plugins, NamedFactory, Interfaces are all OO.
- TBB is GP (so are others like Boost.Pipeline).
- To marry these two paradigms requires a lot of scaffolding (or a better brain than I have).
- Expert input on this very technical detail would be most welcome.

$\rightarrow$ if you have experience/input, let's talk offline!

Port prototype algorithms

- Xin has produced a **huge body of intellectual work** in the prototype.
- Focus was on results, not maintenance or software design.
- Effort is needed to **understand** each algorithm and **refactor** its code into Toolkit concepts.
- **Porters** will contend with:
  - Extracting hard-coded configuration parameters.
  - Monolithic code blocks need to be split.
  - Adapting to data model changes.
  - Conversion of statefull classes to functional ones.
  - Go from "services" to dependency injection.
- Xin has done some **optimization**, but I expect more can be had along the way. Porters should be/get familiar with profiling tools.

Suggest to focus on **reproducing prototype results** before making novel improvements.

Performance Evaluation and Improvements

- Will be done, in part, during the porting exercise.
- But, want a **systematic performance audit** of full chain.
- Understand performance as a function of thread count.
- **Kill any easy bottlenecks**.
- Document before/after performance and the changes.

Maybe not a lot of fun, but the results will make you a hero!

GPU Acceleration

- Best done after CPU-level optimization is exhausted.
- Understand remaining bottlenecks.
- Determine if they can be accelerated with GPU.
- Develop drop-in replacements nodes which run alg on GPU instead of CPU.

This work should be done with awareness of existing and emerging GPU and Phi use on HPC (eg, the new BNL HPC).

Wire Cell Service

Two driving applications:

Bee 2.0 need prompt processing backend services which driven by queries from user on a web browser

HPC (or even Grid/batch) backend driven by queries from a LArSoft client module with Wire Cell processes running on low RAM/core but multi/massive-threaded servers.

- Want reusable components for both applications.
- Tie-ins with next task.
- Best to have prior experience in developing network servers.
- Need to pick a suitable client/server protocol that can be implemented on HPC and can mesh well with C++ and JavaScript clients. Or just use WAMP like with Bee 2.0.

If this is interesting to you? Talk with Chao and me.

Distributed Wire Cell

- Based on Performance Evaluation task, determine if multi-machine parallelism is even required.
- Implement by extending the DFP functionality to support DFP-node connections across the network.
- Need to select one (or better yet allow multiple) network communication methods (MPI, ZeroMQ, WAMP or ???).
- Should work in conjunction with Wire Cell Service work.

As noted, this task is speculative. We may not need it.

Wire Cell on HPC

- Based on Performance Evaluation task, determine if HPC is required or if Wire Cell can keep up on Grid resources alone.
- Porting to HPC requires understanding limitations specific to each target HPC:

Architectural	eg, ROOT doesn't work, dynamic linking considered harmful.
Environmental	eg, special "edge" computing to run LArSoft and/or data source/sink.
Security	HPCs less accessible than Grid nodes. How to get **high data rate** (full raw data) in and results out?
Other	HPC is largely new to neutrino physicists, expect interesting challenges.

Initial efforts:

- Already collaborating with BNL "HPC Code Center".
- BNL has a small, 8 node HPC with GPU for testing now and is building a larger, modern HPC in the coming year.
- Started very useful discussions with ANL to use Mira.
  - link to ANL HPC Training courses

Bee 2.0

Toolkit

Summary

Summary

- Bee 2.0 is an ambitious system allowing **human-directed automatic reconstruction** with follow-on data mining of applied commands.
- Some technical coding/design issues to do but, the **toolkit is soon ready** for outside contributions.
- A number of known tasks are identified and which can largely be worked on in parallel.
- **Volunteers are most welcome!**
  - $\rightarrow$ if interested, let's discuss!
- In general, Wire Cell Toolkit can soon serve as a vehicle to advance **new reconstruction ideas** and apply toward **detector optimization studies**.